Preprint - International Journal of Digital Libraries

Special Issue on "Digital Libraries in Medicine"

Real-Time Digital Libraries Based on Widely Distributed, High Performance Management of Large-Data-Objects

William Johnston, Jin Guojun, Case Larsen, Jason Lee, Gary Hoo, Mary Thompson, Brian Tierney, and Joseph Terdiman, M.D.*

Imaging and Distributed Computing Group, Information and Computing Sciences Division, Ernest Orlando Lawrence Berkeley National Laboratory,[1] Berkeley, CA 94720, USA
* Kaiser Permanente, Division of Research, Oakland, CA 94611, USA

Abstract: We describe a distributed, wide area network based approach to collecting, cataloguing, storing, and providing Web access for large-data-objects that originate as high-speed data streams. Such data streams result from the operation of many types of on-line instruments and imaging systems, and are a "staple" of modern intelligence, scientific, and health care environments. The approach provides for real-time conversion of the data streams and large datasets to "objects" that are manageable, extensible, searchable, browsable, persistent, and available to both "ordinary" and high performance applications through the integration of a high-speed distributed cache and transparent management of tertiary storage. The user interfaces — for both application users and data collection curators — are provided by the capabilities of the World Wide Web.

The capabilities of the architecture are not unlike a digital library system, and we give an example of a digital image library that has been built using this architecture. However, our approach particularly addresses the issues involved in creating such digital library-like collections automatically from the high date-rate, real-time output of, e.g., satellite imaging systems, scientific instruments, health care imaging systems, etc.

We discuss the capabilities, architecture, and implementation of such a system, as well as several example applications in data-intensive environments. The applications include a metropolitan area ATM network based, on-line health care video imaging system, and several image database applications, including a photographic-image library (see [9]). We also describe the security architecture used to enforce data owner imposed access conditions.

Keywords: on-line high data-rate instruments, large-scale storage, high-speed networks, tertiary storage management, digital libraries

1 Introduction

We are evolving a strategy for using high-speed wide-area networks as enablers for widely available high capacity, high performance storage systems, and for the management of data from on-line instruments and imaging systems. The high-level goal is to dramatically increase our ability to capture, organize, search, and provide high-performance and location independent access to "large-data-objects" (LDOs). The data-components of these objects — typically the result of a single operational cycle of a scientific instrument, medical imaging system, or supercomputer run, and of sizes from tens of megabytes to tens of gigabytes — are the staple of modern analytical environments. It is also the case that many of the systems that generate such data-objects are used by a diverse and geographically distributed community — examples from the sciences include physics and nuclear science high energy particle accelerators and detector systems, large electron microscopes, ultra-high brilliance X-ray sources, etc. The health care community has similarly complex imaging and instrumentation systems. In all of these cases, dispersed user communities require location independent access to the data.

In any scenario where data is generated in large volumes and with high throughput, and especially in a distributed environment where the people generating the data are geographically separated from the people cataloguing or using the data, there are several important considerations for managing instrument generated data:

1. The work described in this paper is supported by the U. S. Dept. of Energy, Office of Energy Research, Office of Computational and Technology Research, Mathematical, Information, and Computational Sciences Division (http://www.er.doe.gov/production/octr/mics) and the ER-LTT program, under contract DE-AC03-76SF00098 with the University of California, and by DARPA, Computer Systems Technology Office (http://ftp.arpa.mil/ResearchAreas.html). Contact: wejohnston@lbl.gov, Lawrence Berkeley National Laboratory, mail stop: B50B-2239, Berkeley, CA, 94720, ph: 510-486-5014, fax: 510-486-6363, http://www-itg.lbl.gov). This is report no. LBNL-39613.

- automatic generation of at least minimal metadata;
- automatic cataloguing of the data and the metadata as the data is received (or as close to real time as possible);
- transparent management of tertiary storage systems where the original data is archived;
- facilitation of co-operative research by providing specified users at local and remote sites immediate as well as long term access to the data;
- incorporation of the data into other databases or documents.

The WALDO ("wide-area large-data-object") system is a digital data archive that federates textual and URL linked metadata to represent the characteristics of large data sets. Semi-automatic cataloguing of incoming data is accomplished by extracting associated metadata and converting it into text records, by generating auxiliary metadata and derived data, and by combining these into Web-based objects that include persistent references to the original data-components. Tertiary storage management for the data-components (i.e., the original datasets) is accomplished by using the remote program execution capability of Web servers to manage the data on a mass storage system. For subsequent use, the data-components may be staged to a local disk and then returned as usual via the Web browser, or, as is the case in several of our applications, moved to a high speed cache for access by specialized applications. The location of the data-components on tertiary storage, how to access them, and other descriptive material, are all part of the LDO definition. The creation of object definitions, the inclusion of "standardized" derived-data-objects as part of the metadata, and the use of typed links in the object definition, are intended to provide a general framework for dealing with many different types of data, including, for example, abstract instrument data and multi-component multimedia programs.

WALDO uses an object-oriented approach to provide for capture, storage, catalogue, retrieval, and management of large-data-objects and their associated metadata. The architecture includes a collection of widely distributed services to provide flexibility in managing storage resources, reliability and integrity of access, and high performance access, all in an open environment where the use-conditions for resources and stored information are guaranteed through the use of a strong, but decentralized, security architecture.

The remainder of the paper is organized as follows. In Section 2 we describe the motivation, model and architecture for handling large-data-objects, and the elements, implementation, and applications of that architecture. We also briefly describe the architecture of the Distributed Parallel Storage System — a distributed cache that is used to provide a very high-speed, network-based data cache, and which is a key element of the large-data-object architecture. In Section 3 we describe two applications of the LDO approach. In Section 4 we discuss some of the design issues and their resolution. In Section 5 we briefly describe the security architecture. In Section 6 we describe our experience and the status of the system.

2 Distributed Large-Data-Objects

2.1 Motivation

The advent of shared, widely available, high-speed networks is providing the potential for new approaches to the collection, organization, storage, and analysis of large-data-objects. In one typical example, high-volume health care video and image data used for diagnostic purposes — e.g., X-ray CT, MRI, and cardio-angiography, are collected at centralized facilities and, through the use of the system described here, may be stored, accessed, managed, and referenced at locations other than the point of collection (e.g., the hospitals of the referring physicians).

In health care imaging systems the importance of remote end-user access is that the health care professionals at the referring facility (hospitals or clinics frequently remote from the tertiary imaging facility) will have ready access to not only the image analyst's reports, but the original image data as well. Similarly with data intensive, distributed scientific collaborations, researchers at various sites remote from the data generation and storage require ready access to the data objects. See, e.g., [11] and [7].

The importance of providing and managing *distributed* access to storage is that laboratory instrumentation environments, hospitals, etc., are frequently not the best place to maintain a large-scale digital storage system. Such systems can have considerable economy of scale in operational aspects, and an affordable, easily accessible, high-bandwidth network can provide location independence for such systems.

2.2 Overview

Large-data-objects are a structured collection of metadata, including links to data-components. The metadata typically describes characteristics of the data components, provides access methods information, access control information, etc. Metadata frequently also includes derived data that provide, for example, viewable versions of the data-components. In a sense our LDOs are as much a catalogue entries as objects (since the actual data is usually external to the LDO), but LDOs are intended to be, and are used as, object definitions for large data sets generated by various systems.

The LDO approach provides a mechanism for describing and manipulating "objects" that are the result of federating data-components from many different sources and locations: local files, remote Web sites, remote mass storage systems, etc. Since this work is in the realm of research and development, we have chosen to use the MIME multipart message standard ([17], [18], [19], and [20]) to provide an open and extensible mechanism for object definition. This is a rapidly changing field, and this MIME-based approach may be replaced by the World Wide Web Consortium, Meta Content Framework ([2]).

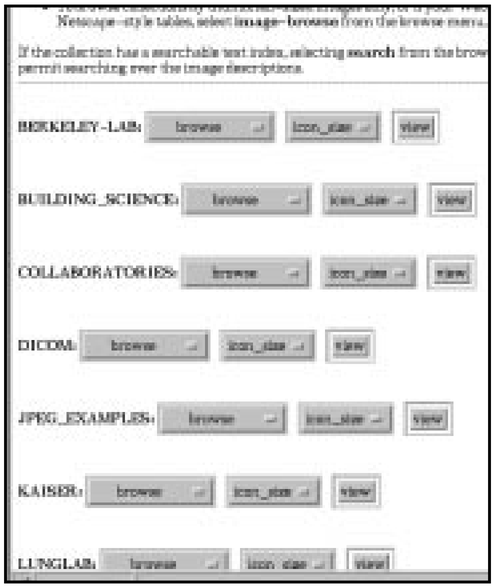
The elements of our large-data-object model include generalized object definition (a class); groups of associated objects called "collections"; and access, search, data entry and security methods for managing the object definition. The object definition is manifested as a Web document that contains metadata in the form of text and typed links to associated objects (e.g., derived information), and typed links to the original large-data-component locations. The typed links are generally URLs with an associated MIME type (see [17]). The typing allows a user or application to locate appropriate data access methods in advance of requesting a (potentially large) data component. Each of the object components can typically be referenced as an individual Web entity, and located, accessed, and manipulated through Web servers using the HTTP protocol. Applications can use the HTTP protocol to request data components, and users can access data directly via Web browsers. Data may be returned directly to the browser or to an application that can use the access method specified by the corresponding MIME type. In the case of high-performance applications, handles to the data on the distributed high-speed cache are returned to the application.

A typical user interface to a large-data-object class using this model can be illustrated by a collection of high resolution microscope images that form a digital-photo collection (see `http://imglib.lbl.gov`). In this case, the images are organized into "sub-collections" (an associated set of objects of one class) partly for ease of browsing and partly to facilitate curation. The object class (collection) has associated search methods, object access methods, and security methods to enforce use-conditions. A Web browser accesses an LDO component via a URL request that typically invokes a Web server cgi-script that implements the necessary methods. The user requests may be for searching, displaying some or all of the LDO components, or modifying the LDO components (collection curation). The digital images in this collection are generated by an off-line scanning electron microscope and are loaded into the Lab's mass storage system by bulk file transfer. Sets of images, representing several days' work are entered into the Image Library with one operation. The textual metadata that is common to a whole set is preloaded and copied automatically to each image, the thumbnails are generated automatically. Later the curator may add additional metadata for specific images. An example of automatic LDO generation from an on-line instrument is described later in this paper.

2.2.1 WALDO Example

To illustrate the WALDO approach to object libraries, we will use a research image collection to illustrate the various user interfaces and the overall model, as well as the structure of large-data-objects.

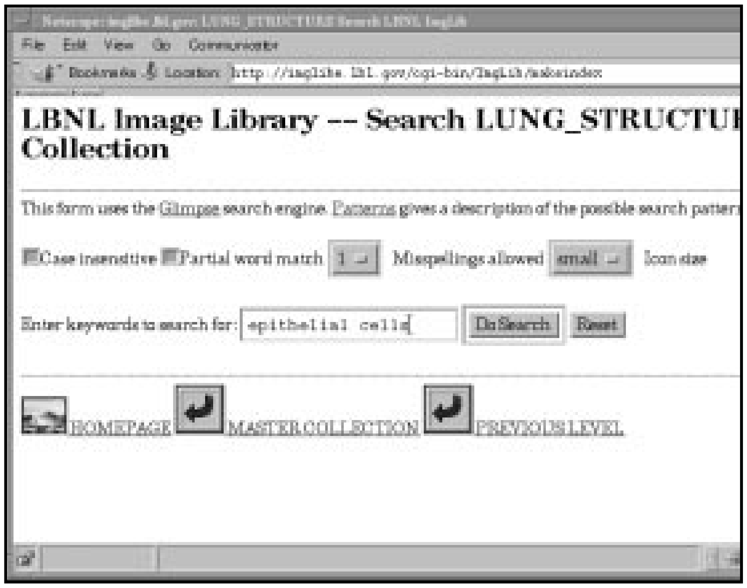
The image library user typically first enters the top level collection index (Figure 1) and initiates a search (on, e.g., the textual metadata as in Figure 2). The textual metadata search method is provided by a modified version of glimpse [6]. The result of the approximate search specified as in Figure 2 is illustrated in Figure 3. The LDOs that satisfy the


Netscape-style tables, select **image-browse** from the browse menu.

If the collection has a searchable text index, selecting **search** from the brow
permit searching over the image descriptions.

BERKELEY-LAB:
BUILDING_SCIENCE:
COLLABORATORIES:
DICOM:
JPEG_EXAMPLES:
KAISER:
LUNGLAB:

Figure 1. A top level collection index.

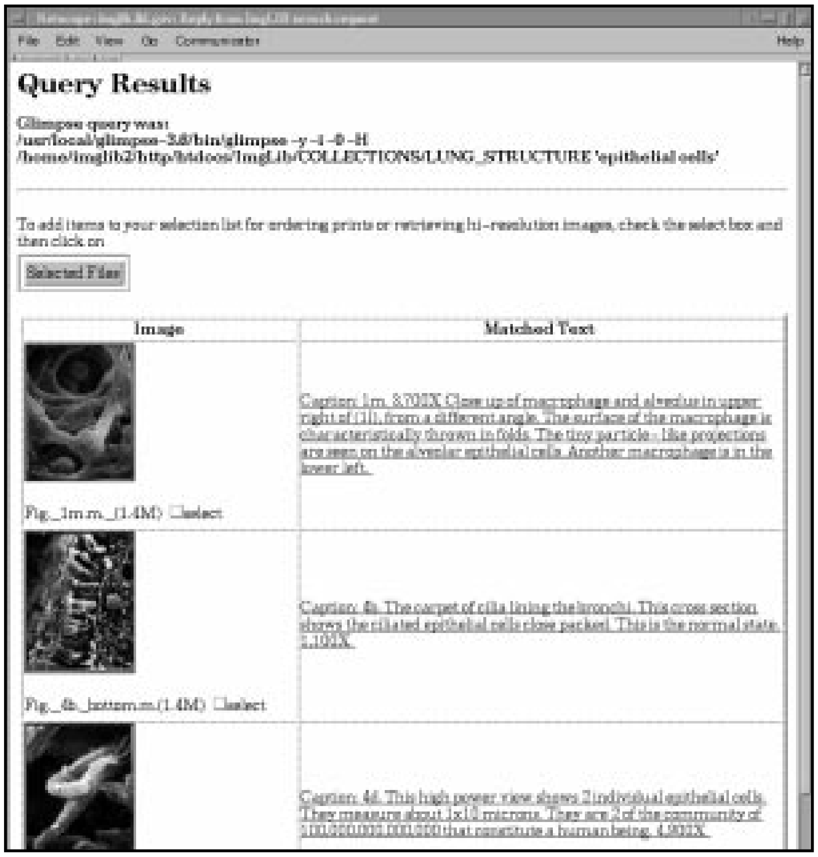

File Edit View Go Communicator
Bookmarks Location: http://imglibs.lbl.gov/cgi-bin/ImgLib/mdsindex
**LBNL Image Library -- Search LUNG_STRUCTU
Collection**

This form uses the Glimpse search engine. Patterns gives a description of the possible search patter

Case insensitive Partial word match 1 Misspellings allowed small Icon size

Enter keywords to search for: epithelial cells
Do Search
Reset
HOMEPAGE
MASTER COLLECTION
PREVIOUS LEVEL

Figure 2. The collection metadata search form.



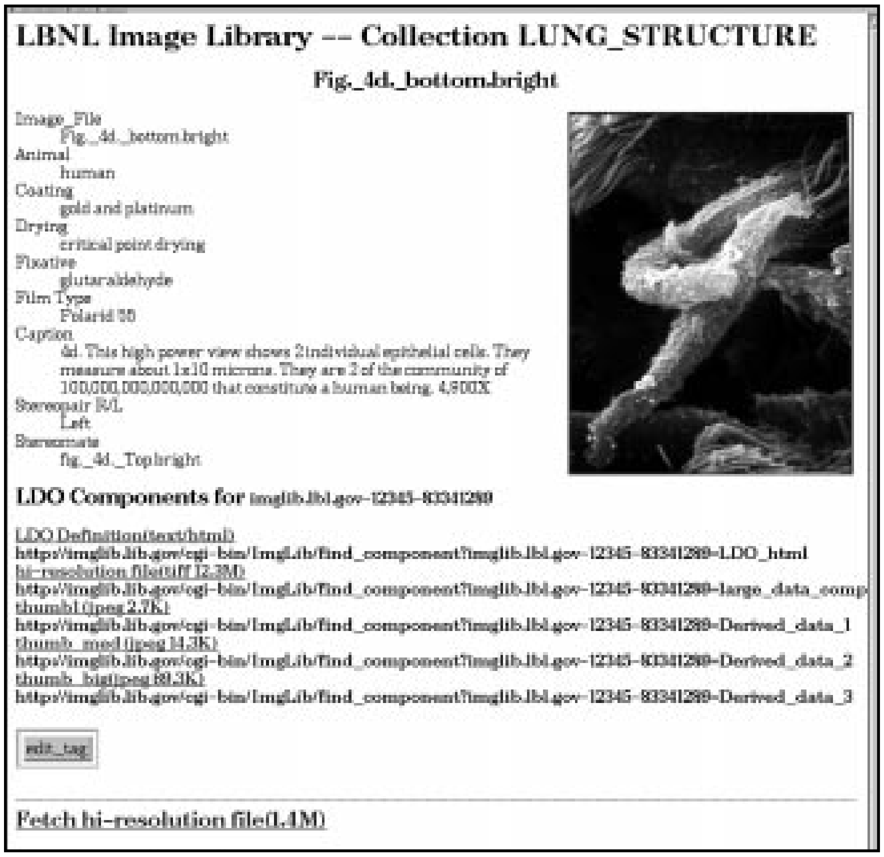
Query Results

Glimpse query was:
/usr/local/glimpse-3.8/bin/glimpse -y -i -0 -H
/home/imglib2/http/htdocs/ImgLib/COLLECTIONS/LUNG_STRUCTURE 'epithelial cells'

To add items to your selection list for ordering prints or retrieving hi-resolution images, check the select box and then click on

Image	Matched Text
Fig._1m.m._(1.4M) ☐select	Caption: 1m. 8,700X Close up of macrophage and alveolus in upper right of (1l), from a different angle. The surface of the macrophage is characteristically thrown in folds. The tiny particle - like projections are seen on the alveolar epithelial cells. Another macrophage is in the lower left.
Fig._4b._bottom.m.(1.4M) ☐select	Caption: 4b. The carpet of cilia lining the bronchi. This cross section shows the ciliated epithelial cells close packed. This is the normal state. 1,100X.
	Caption: 4d. This high power view shows 2 individual epithelial cells. They measure about 1x10 microns. They are 2 of the community of 100,000,000,000,000 that constitute a human being. 4,900X.

Figure 3. The (partial) results of the search specified in the form illustrated in Figure 2.



LBNL Image Library -- Collection LUNG_STRUCTURE

Fig._4d._bottom.bright

Image_File
 Fig._4d._bottom.bright
Animal
 human
Coating
 gold and platinum
Drying
 critical point drying
Fixative
 glutaraldehyde
Film Type
 Polarid 55
Caption
 4d. This high power view shows 2 individual epithelial cells. They measure about 1x10 microns. They are 2 of the community of 100,000,000,000,000 that constitute a human being. 4,900X
Stereopair R/L
 Left
Stereomate
 fig._4d._Top bright

LDO Components for imglib.lbl.gov-12345-83341289

LDO Definition(text/html)
https://imglib.lib.gov/cgi-bin/ImgLib/find_component?imglib.lbl.gov-12345-83341289-LDO_html
hi-resolution file(tiff 12.3M)
https://imglib.lib.gov/cgi-bin/ImgLib/find_component?imglib.lbl.gov-12345-83341289-large_data_comp
thumb1 (jpeg 2.7K)
https://imglib.lib.gov/cgi-bin/ImgLib/find_component?imglib.lbl.gov-12345-83341289-Derived_data_1
thumb_med (jpeg 14.3K)
https://imglib.lib.gov/cgi-bin/ImgLib/find_component?imglib.lbl.gov-12345-83341289-Derived_data_2
thumb_big (jpeg 89.3K)
https://imglib.lib.gov/cgi-bin/ImgLib/find_component?imglib.lbl.gov-12345-83341289-Derived_data_3

Fetch hi-resolution file(1.4M)

Figure 4. The Web browser manifestation of an LDO definition document.

search are represented as a set of "thumbnails" (derived data elements), pointers to the primary data elements (e.g., "Fig._4d._bottom.bright" — a 1.4 Mbyte TIFF-image file in this case), and the "context" in the tag-field in which glimpse found the search term. The displayed context links to the LDO definition document (Figure 4). The thumbnail image in the LDO document links to (what is for this collection) the primary derived data element (a screen-size image, e.g., Figure 5). Figure 6 illustrates one of the browser interfaces. The ability to both search and browse based on the metadata are important elements of our digital library model.

For the data-components that reside only on tertiary storage (e.g., a tape-robot-based mass storage system), there is an option for forcing migration of these LDO components back to an on-line cache. The interface for managing the tertiary storage (e.g., a hierarchical mass storage system) is illustrated in Figure 7.

Another aspect of the model is that there are derived-data-components that are viewable with "standard" Web browsers. The original data-components — very large, high resolution, full color TIFF format images in the case of the digital-photo library or DICOM images in the case of medical libraries — are typically not directly viewable.

2.3 Functionality Goals

Our model for the capabilities of a distributed large-data-object system includes the following elements:

- real-time cataloguing of extensible, linked, multi-component data-objects that can be asynchronously generated by remote, on-line data sources
- class-based methods for management of the large-data-objects
- on-line metadata, with cacheable off-line components.
- representation of the object components as Web-accessible elements
- explicit association of access methods with the data components
- flexible curator / collection-owner management of collections of data-objects, including "any-time" management of the collection organization and object metadata
- globally unique and persistent naming of the objects and their various components via URLs and URNs
- strong access control at the level of individual object components based on use-condition certificates managed by the data owner
- high-performance application access to the data-components
- flexible and extensible approaches to searching.

2.4 LDO Structure, and WALDO Data Model and Software Architecture

Our model for "objects" includes:

- ◆ A large-data-object (LDO), consisting of
  - a standard set of identifying information (e.g., class, unique id, owner, collection name)
  - one or more typed links to the original data-components
  - one or more typed links to derived data-components
  - tags defining access control in terms of pointers to the entities who set access conditions and pointers to the servers where access condition certificates are located
  - class-specific textual metadata
- ◆ An LDO is an instantiation of a class prototype. Each LDO contains a subset of the data elements defined for its class. An LDO is manifested as a persistent Web document, and class browsers can display some or all of its components. Small components, e.g., text and thumbnail images, may be directly displayed by browsers. Data-components are referenced by a persistent URL. An LDO class is provided with
  - search methods
  - data component access methods
  - data entry methods
  - access control methods
- ◆ Collections consist of LDO's and hierarchically organized sub-collections of LDOs in that class
- ◆ Class / collection roles
  - collection owner
    - establishes agreements with the WALDO Web server for resources
    - defines the LDO classes for the collection
    - defines the basic access control parameters, including potential delegation of authority for setting access conditions
  - curators are users with write access to objects, and can
    - add, delete, and modify LDOs or components of LDOs
    - manage the collection structure (can create sub-collections and may move LDOs between sub-collections)

Figure 8 illustrates the data flow and overall organization of the WALDO architecture. It also indicates the central role of high-speed cache which is used both for initial data collection, and to provide subsequent high-speed access by applications.

The basic elements of the architecture (referring to Figure 8) include:

- data collection systems and the instrument-network interfaces (1)

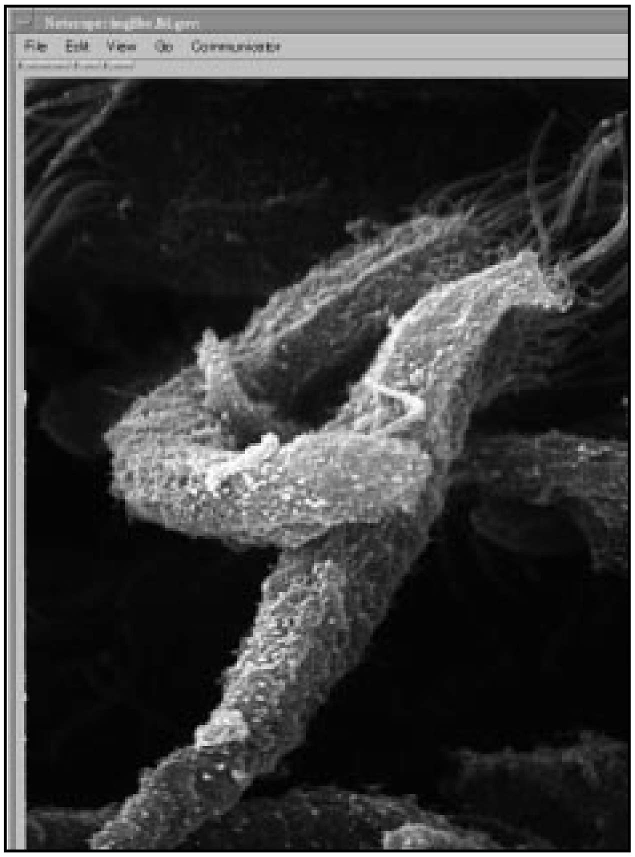


Figure 5. A typical object representation that is automatically derived from the original data when the LDO

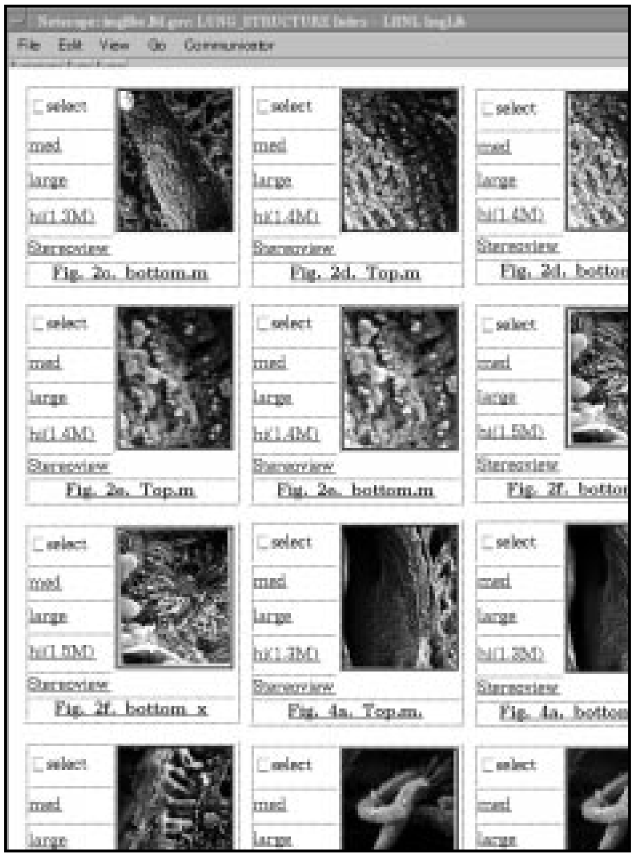
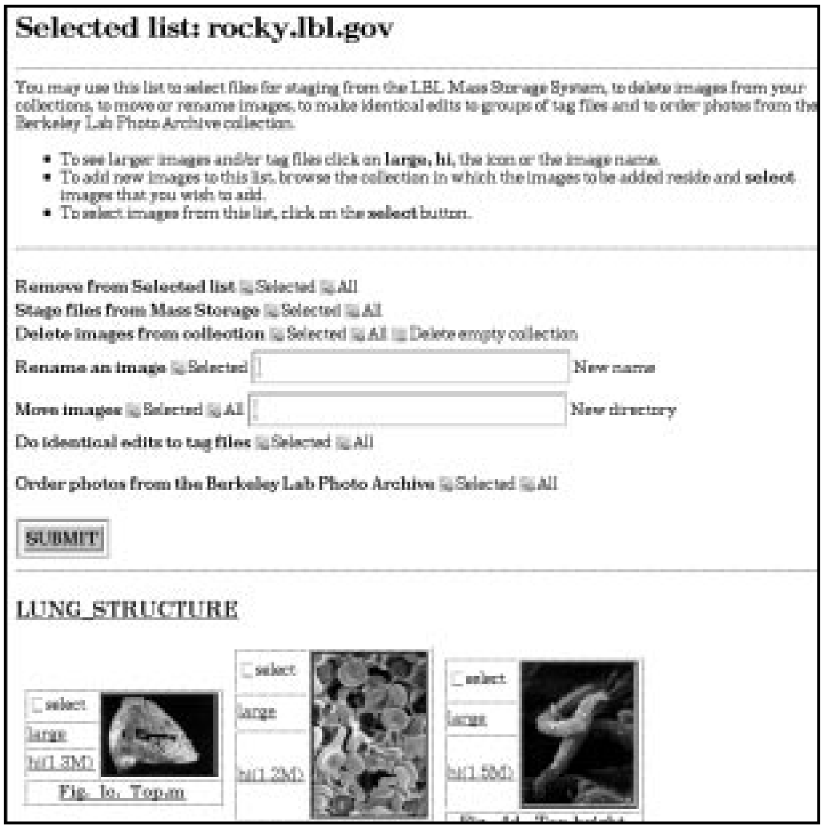


Figure 6. The result of browsing the collection that contained one of the results of the query illustrated in Figure 2.



Selected list: rocky.lbl.gov

You may use this list to select files for staging from the LBL Mass Storage System, to delete images from your collections, to move or rename images, to make identical edits to groups of tag files and to order photos from the Berkeley Lab Photo Archive collection.

- To see larger images and/or tag files click on **large, hi,** the icon or the image name.
- To add new images to this list, browse the collection in which the images to be added reside and **select** images that you wish to add.
- To select images from this list, click on the **select** button.

Remove from Selected list Selected All
Stage files from Mass Storage Selected All
Delete images from collection Selected All Delete empty collection
Rename an image Selected New name
Move images Selected All New directory
Do identical edits to tag files Selected All
Order photos from the Berkeley Lab Photo Archive Selected All

Figure 7. The tertiary storage system management interface. Data-components can be selectively migrated from archival storage back to on-line storage.

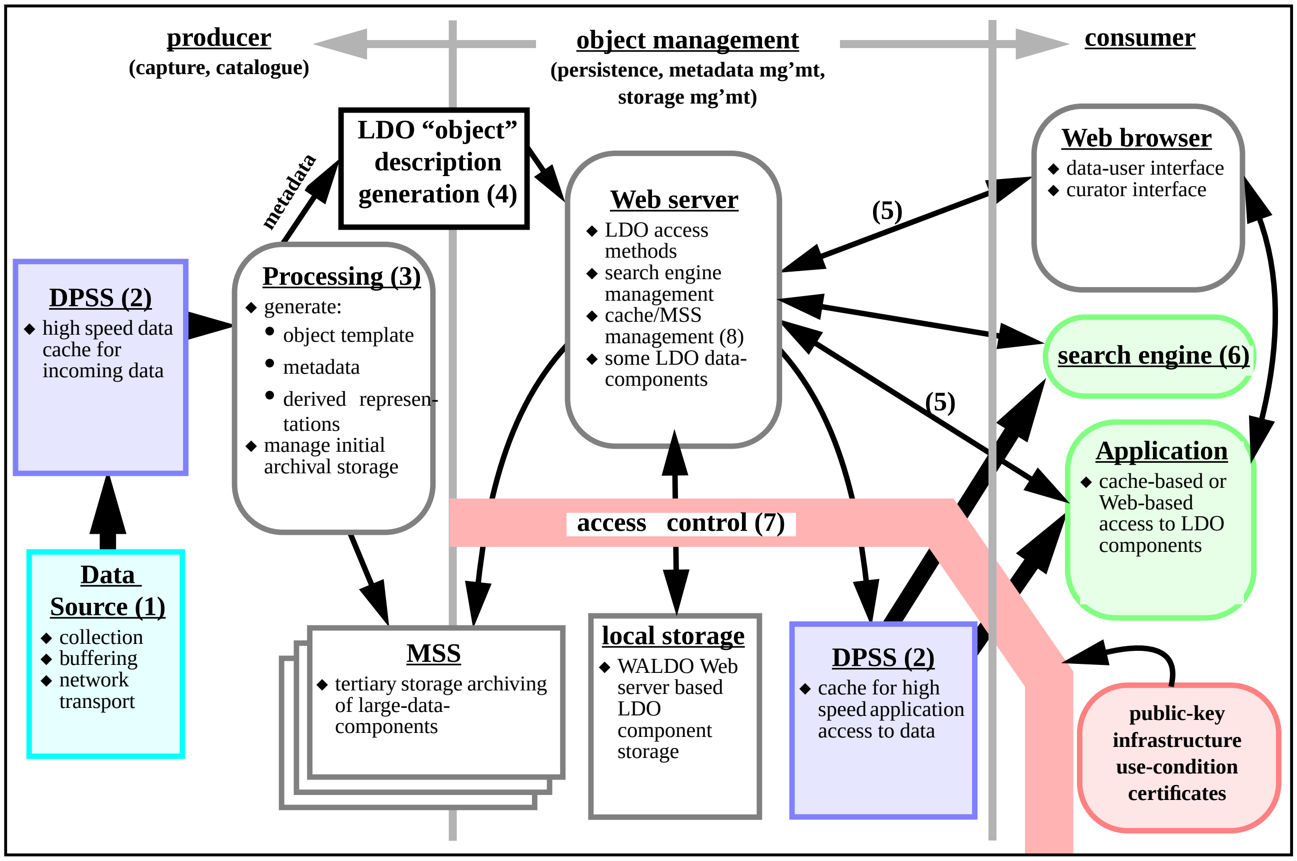

producer
(capture, catalogue)
object management
(persistence, metadata mg'mt, storage mg'mt)
consumer
LDO "object" description generation (4)
metadata
Web server
◆ LDO access methods
◆ search engine management
◆ cache/MSS management (8)
◆ some LDO data-components
Web browser
◆ data-user interface
◆ curator interface
(5)
DPSS (2)
◆ high speed data cache for incoming data
Processing (3)
◆ generate:
• object template
• metadata
• derived representations
◆ manage initial archival storage
search engine (6)
(5)
Application
◆ cache-based or Web-based access to LDO components
access control (7)
Data Source (1)
◆ collection
◆ buffering
◆ network transport
MSS
◆ tertiary storage archiving of large-data-components
local storage
◆ WALDO Web server based LDO component storage
DPSS (2)
◆ cache for high speed application access to data
public-key infrastructure use-condition certificates

Figure 8. The distributed Large-Data-Object overall architecture and data flow.

- high-speed, network-based cache storage for receiving data, for providing intermediate storage for processing, and for high-speed application access (2)
- transparent tertiary storage ("mass storage") management for the data-components (8)
- processing mechanisms for various sorts of data analysis and derived data generation (3)
- data management that provides for the automatic cataloguing and metadata generation that produces the large-data-object definitions (4)
- data access interfaces, including application-oriented interfaces (5)
- curator interfaces for managing both the metadata and the LDO collection organization
- user access interfaces for all relevant aspects of the data (applications, data, and metadata)
- flexible mechanisms for providing various searching strategies (6)
- transparent security that provides strong access control for the data components based on data-owner policies (7)

These elements are all provided with flexible, location-independent interfaces so that they can be freely (transparently) moved around the network as required for operational or other logistical convenience.

2.4.1 The Distributed-Parallel Storage System—A High-Performance Network Cache

A high performance, widely distributed, network storage system is an essential component of a network-based large-data-object environment. Within the high performance network cache — a "middleware" service — the architectural issues include:

- highly distributed and parallel cache operation
- distributed, autonomous component management
- user access methodologies
- security architecture

Distributing the components of a storage system throughout the network can increase its capacity, reliability, performance, and security. Usable capacity increases through the aggregation of distributed storage components. Highly reliable access can be provided through redundancy of data and storage systems that are configured from components that have little in common (e.g., location). Performance is increased by the combined characteristics of parallel operation of many sub-components, and the independent data paths provided by a large network infrastructure. Security is potentially increased by having many independent components, each of which has local and independent enforcement mechanisms that can limit the scope of a security breach.

The Distributed-Parallel Storage System ("DPSS") is an experimental system in which we are developing, implementing, and testing these ideas. In most configurations, the DPSS is used as a network-based, high performance, random access logical block server designed to supply and consume high-speed data streams for other network-based processing systems. (See [21].)

The DPSS is a "logical block" server whose functional components are distributed across a wide-area network. (See Figure 9.) The DPSS is fundamentally a random-access logical block server: There is no inherent organization to the blocks, and in particular, they would never be organized sequentially on a server. The data organization is determined by the application as a function of data type and access patterns so that a large collection of disks and servers can operate in parallel enabling the DPSS to perform as a high-speed data source or data sink.

At the application level, the DPSS provides a semi-persistent cache of named data-objects, and at the storage level it is a logical block server. Although not strictly part of the DPSS architecture, the system is usually provided with application libraries that implement data set access methods. These provide object-like encapsulation of the data in order to represent complex user-level data structures so that the application does not have to retain this information for each different data set. Data access methods currently include simple video data access and the more complex access methods required by TerraVision and STAR.[2] The access method converts the application requests into logical block requests. These logical block requests are then sent to the DPSS Master which serves two functions: logical block name translation and resource management. Resulting physical block requests are forwarded to disk servers that return data directly to the application using parallel data paths.

3 Applications

We currently have two example applications of the LDO architecture. One is a metadata-based digital image library system that is oriented toward curation. That is, the structure and tools of the system are intended to provide for easy and

2. TerraVision is an interactive terrain navigation application using multi-resolution, tiled, image pyramids [13]. STAR is an external data representation access (XDR) method [7].

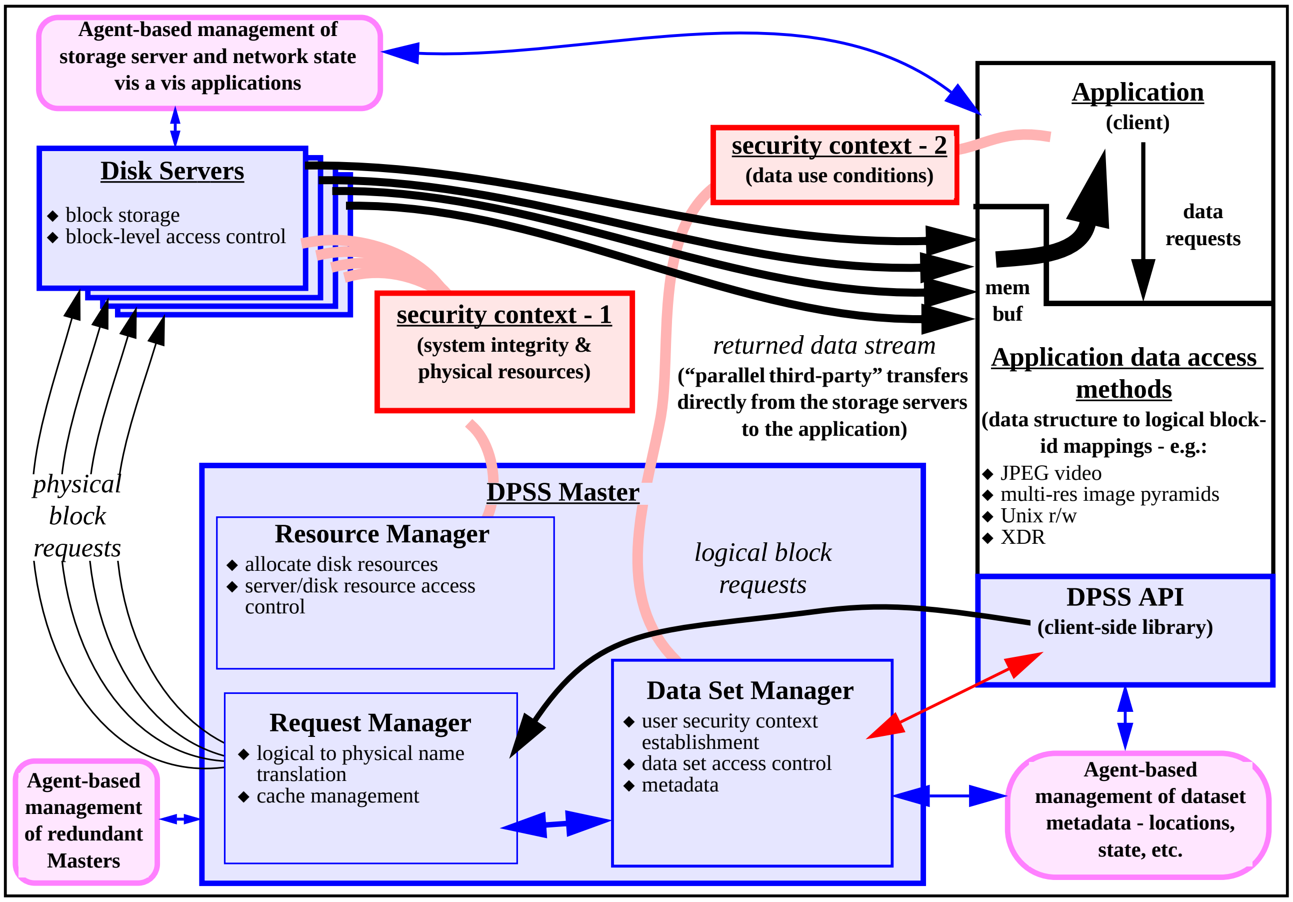

Agent-based management of storage server and network state vis a vis applications
Disk Servers
◆ block storage
◆ block-level access control
security context - 2
(data use conditions)
Application
(client)
data requests
mem buf
security context - 1
(system integrity & physical resources)
returned data stream
("parallel third-party" transfers directly from the storage servers to the application)
Application data access methods
(data structure to logical block-id mappings - e.g.:
◆ JPEG video
◆ multi-res image pyramids
◆ Unix r/w
◆ XDR
physical block requests
DPSS Master
Resource Manager
◆ allocate disk resources
◆ server/disk resource access control
logical block requests
DPSS API
(client-side library)
Request Manager
◆ logical to physical name translation
◆ cache management
Data Set Manager
◆ user security context establishment
◆ data set access control
◆ metadata
Agent-based management of redundant Masters
Agent-based management of dataset metadata - locations, state, etc.

Figure 9. The Distributed-Parallel Storage System architecture.

continuous maintenance by the data owners. The second example is a real-time data capture and cataloguing system that is oriented toward the automated capture of data from on-line instruments and the creation of digital libraries on-the-fly.

3.1 Curator Managed Collections: Digital-Image Libraries

Digital image libraries require flexible curator management functions so that curators from a variety of backgrounds can easily maintain their own collections. One user community is scientists who maintain annotated image collections that are associated with their scientific laboratory environment. These collections are used like a laboratory notebook, and are continuously updated. See, for example (as in Figure 6) `http://imblib.lbl.gov/cgi-bin/ImgLib/displaytag/LUNG_DEMO/tags/Fig.2?both=small`) .

Another use of this flavor of the system is the "Berkeley Lab On-line Photo Archive" (`http://imglib.lbl.gov/ImgLib/photo-archive.html`). This application is essentially the same as the annotated scientific image collection, but the use characteristics are somewhat different. In particular, this collection undergoes a lot of rearrangement as it grows and the subject matter diversifies. Persistent references are also very important in this collection, as the intent is for many other documents to use this collection to provide their images via links (e.g., the "LBL Newsmagazine, Fall 1981: 50th Anniversary Issue" at `http://imglib.lbl.gov/LBNL_Res_Revs/50th`). This has raised some interesting design issues that are discussed in Section 4.

3.2 Automatically Generated Collections: Real-Time Capture and Cataloguing of Data from On-line Instruments

The model for real-time capture and cataloguing is that a data source generates units of data that have associated auxiliary information. As the data is processed in real-time, the "units" can be identified as such, and the appropriate associated data can also be identified.

A key issue with "identification" of a unit of data is to know when the unit has been completely received so that cataloguing and processing to create an LDO can commence. This identification can take the form of explicit in-band markers in the data stream, or out-of-band notification that all of the data of one unit has been sent, or the identification could be implicit, such as a data unit being defined by a timing mechanism (e.g., one data unit is whatever is received in some specified period of time) or by a quantity mechanism (e.g., one data unit is a fixed quantity of data).

Likewise, the acquisition of associated data (metadata pre-cursors) can be via explicit in-band or out-of-band separate data units, or implicit (e.g., the timestamp of a data unit whose boundary is defined by a point in time).

3.2.1 Video Data: On-line Cardio-angiography

An example of a medical application that uses the automatically managed approach is a system that provides for collection, storage, cataloguing, and playback of video-angiography data.[3]

> *"Cardio-angiography is used to monitor and restore coronary blood flow, and though clinically effective, the required imaging systems and associated facilities are expensive. To minimize the cost of such procedures, health care providers are beginning to concentrate these services in a few high-volume tertiary care centers. Patients are typically referred to these centers by cardiologists operating at clinics or other hospitals; the centers then must communicate the results back to the local cardiologists as soon as possible after the procedure. The advantages of providing specialized services at distant tertiary centers are significantly reduced if the medical information obtained during the procedure is not delivered rapidly and accurately to the referring physician at the patient's home facility. The delivery systems currently used to transfer patient information between facilities include interoffice mail, U.S. Mail, fax machine, telephone, and courier. Often these systems are inadequate and potentially could introduce delays in patient care." (From [12].)*

Using a shared, metropolitan area ATM network, and a high-speed distributed data handling system, video sequences are collected from the video-angiography imaging system, then processed, catalogued, stored, and made available to remote users. This permits the data to be made available in near real-time to remote clinics (see Figure 10). The LDO becomes available as soon as the catalogue entry is generated — derived data is added as the processing required to produce it completes. Whether the storage systems are local or distributed around the network is entirely a function of the optimal logistics.

This application was developed in the Kaiser CalREN project, a joint project of Lawrence Berkeley National Laboratory, Kaiser Permanente, Philips Palo Alto Research center, and the Pacific Bell CalREN program [12]. Angiography data is collected directly from a Philips scanner by a computer system in the San Francisco Kaiser hospital Cardiac Catheterization Laboratory. This system is, in turn, attached to an ATM network provided by the National Transparent Optical Network testbed (NTON). NTON provides OC-3 (155 Mbits/s) and OC-12 (622 Mbits/s) connections to a very high-speed backbone (see [15]). When the data collec-

3. Cardio-angiography imaging involves a two plane, X-ray video imaging system that produces from several to tens of minutes of digital video sequences for each patient study for each patient session. The digital video is organized as tens of data-objects, each of which are of the order of 100 megabytes.

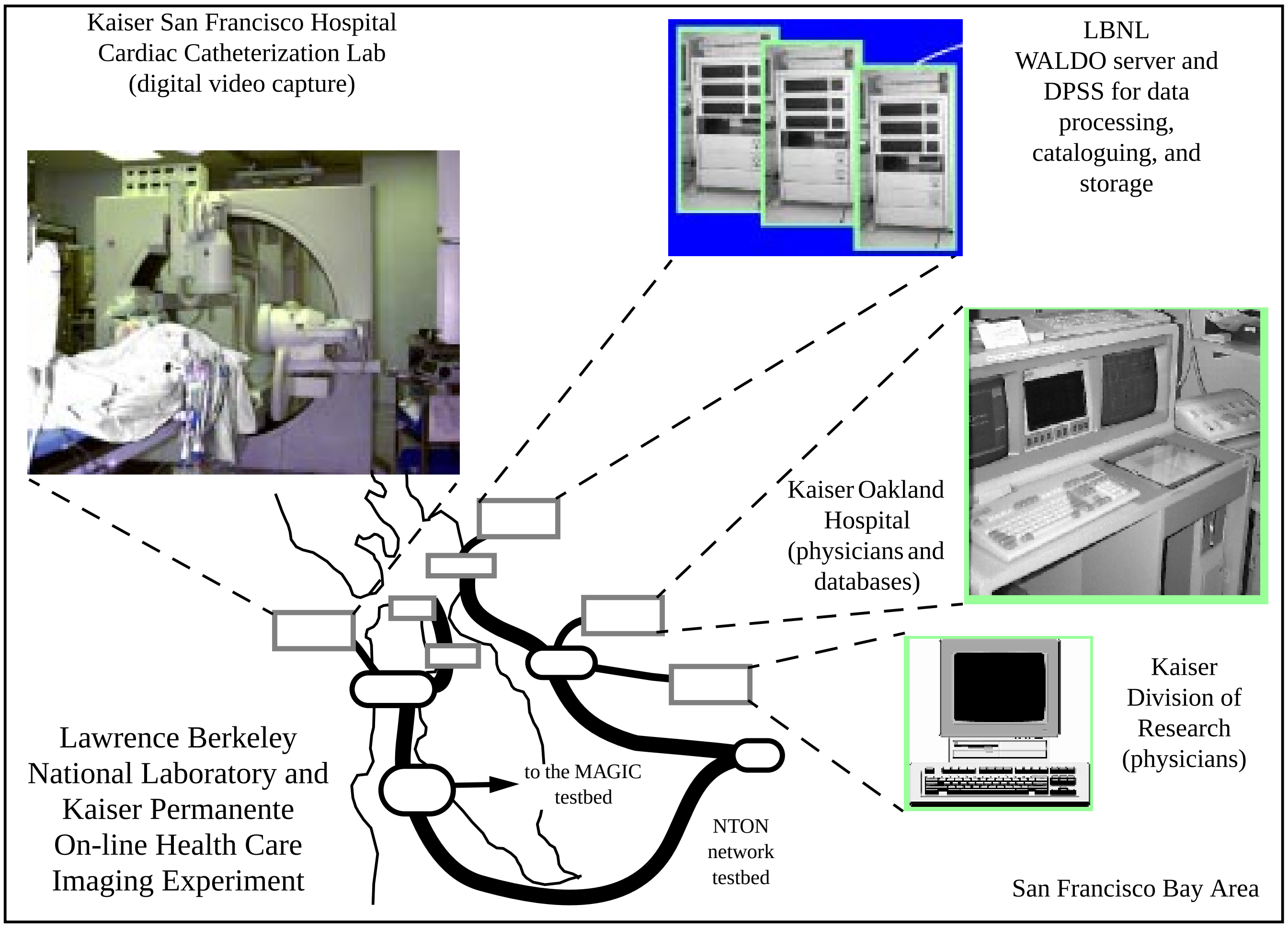


Figure 10. Physical architecture of the health care imaging application as it is embedded in the National Transparent Optical Network testbed.

tion for a patient is complete (about once every 20–40 minutes), 500–1000 megabytes of digital video data is sent across the ATM network to LBNL (in Berkeley) and stored first on the DPSS, and then the LDO object definitions are generated and made available via the Web. Auxiliary processing and archiving to one or more mass storage systems proceeds independently. This process goes on 8–10 hours a day.

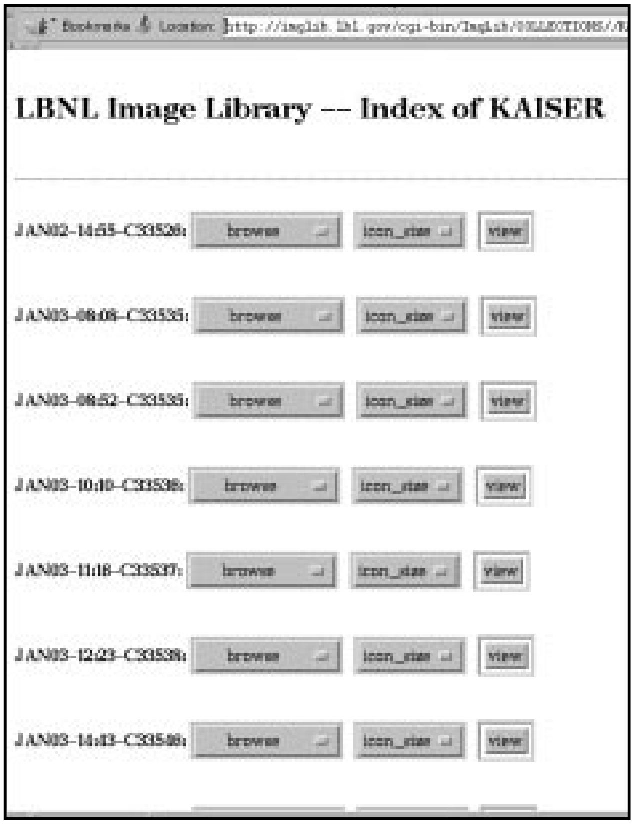
The WALDO management tools provide the Web based user interface to the data, and to appropriate viewing applications. Hospital department-level Web-based patient databases can then refer directly to the data in WALDO without duplicating that data, or being concerned about tertiary storage management (which is handled by WALDO).

Figures 11-13 illustrate the user environment provided by the WALDO model for this application.

Figure 11 illustrates browsing the top-level of the angiography collection. A “year” or “hospital-of-patient-origin”, or both, will probably eventually be added to the collection hierarchy above these individual study pointers as part of the collection curation. If this is done, then the persistence mechanisms described in Section 4 will come into play so that links created while the collection has its current organization will remain valid.

Figure 12 illustrates browsing a collection of LDOs (in this case, related by the fact that they are all for a single patient study). At the top of the page is a brief description of the collection (the study identifiers in this case), some standard instructions, and the tertiary storage migration interface. This is followed by the two special applications that understand the bi-plane video format. These are invoked from the Web page, and the normal mode of operation is to view all of the video segments of a study in sequence. When these viewers are invoked they are also given all of the pointers to the original dataset components of the LDO. To facilitate access from unmodified Web browser environments, one of derived data types is an MPEG coding of the video.

Figure 13 illustrates the result of invoking the bi-plane video player that accesses data from the network cache.

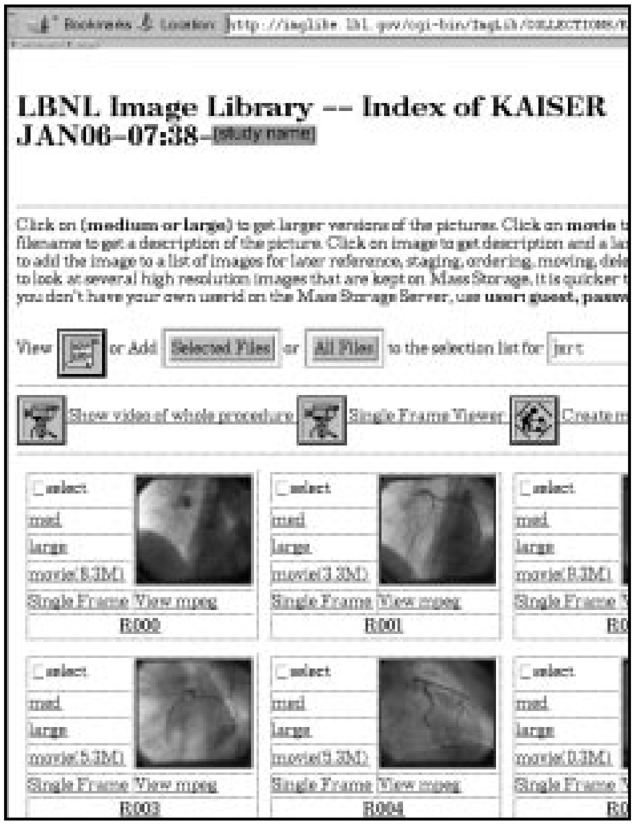


http://imglib.lbl.gov/cgi-bin/ImgLib/COLLECTIONS/K

LBNL Image Library -- Index of KAISER

JAN02-14:55-C33526:
JAN03-08:08-C33535:
JAN03-08:52-C33535:
JAN03-10:10-C33536:
JAN03-11:18-C33537:
JAN03-12:23-C33538:
JAN03-14:13-C33540:

Figure 11. An index built from incoming angiography video data in real-time.



http://imglibe.lbl.gov/cgi-bin/ImgLib/COLLECTIONS/K

LBNL Image Library -- Index of KAISER JAN06-07:38-[study name]

Click on (**medium or large**) to get larger versions of the pictures. Click on **movie**
filename to get a description of the picture. Click on image to get description and a la
to add the image to a list of images for later reference, staging, ordering, moving, dele
to look at several high resolution images that are kept on Mass Storage, it is quicker t
you don't have your own userid on the Mass Storage Server, use **user: guest, passw**

View or Add Selected Files or All Files to the selection list for jart

Show video of whole procedure Single Frame Viewer Create m

movie(8.3M) R000
movie(3.3M) R001
movie(8.3M)
movie(5.3M) R003
movie(9.3M) R004
movie(0.3M)

Figure 12. The browser interface for a sub-collection that represents one patient study (multiple video clips).

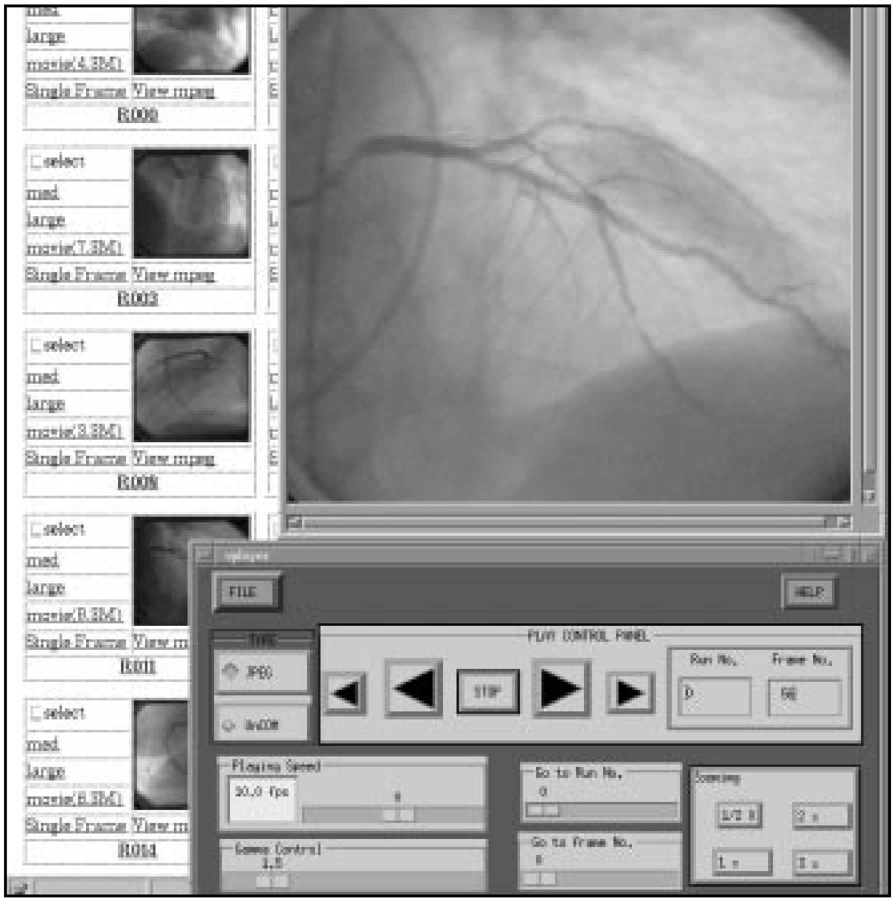


Figure 13. The user interface of the special application that displays the bi-planar video-objects directly from the DPSS cache.

4 Experience and Design Issues

Probably the predominant characteristic of cataloguing live data sources is that the LDO components — data-components, metadata and derived data — show up asynchronously as a natural result of the automated processing of the data-components and metadata that go into making up an LDO. To accommodate automated processing, and meet the capability objectives, several different methods of representing, storing, and presenting the information resulting from LDO generation have been tried in the course of the evolution of the LDO implementation. The approach that has addressed several issues is to have the various Web accessible representations of LDOs and their components be dynamically generated by a collection of object management methods. These methods typically scan the collection for new object components, and generate an html document that presents those components (e.g., the object browser illustrated in Figure 12).

4.1 Dynamic indexing

Since the complete large-data-object is produced over as much as several days (some of the processing for the derived files — e.g., MPEG coding of video — can be very time consuming) a natural way to produce the Web representations of the LDO definitions is to generate them on-the-fly. For example, the LDO definition document shown in Figure 4 was generated using the LDO access method "displaytag", invoked by the Web invocation:

```
http://imglib.lbl.gov/cgi-bin/ImgLib/
displaytag/LUNG_STRUCTURE/tags/
Fig_4d._bottom.bright                    (1)
```

Figure 6 illustrates a dynamically created browsable index for the "Lung_Structure" collection of high-resolution digital microscope images. This index is generated by a single Web cgi invocation that looks like:

```
http://imglib.lbl.gov/cgi-bin/ImgLib/
makeindex?LUNG_STRUCTURE=browse          (2)
```

(In other words, the "`makeindex`" program is invoked with arguments "`LUNG_STRUCTURE`" and "`browse`" to index this sub-collection.)

This technique provides the curator with the flexibility to reorganize the collections and be assured that the browsable indexes the user sees are always up-to-date.

This same general approach is used for most Web-based manipulation of LDOs. However, while this approach provides several of the desired capabilities, it does not, on the face of it, provide the required persistence characteristics.

4.2 Persistence

There are several aspects of persistence that we would like to provide, including global and easily usable names for both the LDOs and their primary components.

Naming uniqueness, and persistence (up to, but not including, the Internet DNS name of the WALDO Web server), can be provided by including a globally unique object identifier (OID) in each LDO. Our approach involves several points.

First, we generate an OID during the LDO creation that is of the form:

```
Internet_DNS_name-process_id-
time_of_day_in_nanoseconds                (3)
```

Second, as one of the LDO methods, we have provided the capability to locate and refer to an LDO component:

```
find_component (OID, component_name)      (4)
```

Which can be invoked as a cgi request:

```
http://imglib.lbl.gov/cgi-bin/ImgLib/
find_component?<OID>&<component_name>     (5)
```

Ignoring, for the moment, the general issue of LDO component naming, at least one standard invocation of this function, e.g.,

```
find_component (OID,'LDO-html')           (6)
```

will always return an html version of the LDO definition corresponding to the given OID. (E.g., Figure 4.)

With standard tags (or their aliases), the `find_component` method can perform a search of OID-space and return the requested component based on the tag. For example

```
http://imglib.lbl.gov/cgi-bin/ImgLib/
find_component?<OID>&thumbnail            (7)
```

obtains the "thumbnail" derived visual for any LDO in those classes that use the thumbnail concept.

To provide an "easily used" persistent reference we build a permanently instantiated (but periodically updated) index document at the collection level. The index entries consist of the collection name (which can change as the collection is reorganized), a short LDO description, and the text of the persistent URL (e.g., (5), above) that will return the html version of the LDO. For example, the index entry

```
BERKELEY-LAB HISTORY SEABORG-
PERSPECTIVE JFK visiting Lawrence
Radiation Laboratory, March 23, 1963
```

```
(OID=imglib.lbl.gov:123426:83341287767
634000)

  URL= http://imglib.lbl.gov/cgi-bin/
ImgLib/find_component?imglib.lbl.gov-
12342683341287767634000&LDO-html         (8)
```

is the text string defining the name of the underlying link:

```
<a href="http://imglibe/ImgLib/
COLLECTIONS/BERKELEY-LAB/HISTORY/
SEABORG-PERSPECTIVE/index/
96B05392.html">
BERKELEY-LAB HISTORY SEABORG-
PERSPECTIVE JFK visiting Lawrence
Radiation Laboratory, March 23, 1963
(OID=imglib.lbl.gov:123426:83341287767
634000)URL= http://imglib.lbl.gov/cgi-
bin/ImgLib/
find_component?imglib.lbl.gov-12346-
8334128776763400O&LDO-html)</a>          (9)
```

As an optimization, the actual links may point to a “local” copy of the html version of the LDO metadata, rather than to the cgi invocation that searches for and returns the “real” metadata (i.e., the query represented by the version of the URL in (8)). Both the html document that is the index set, and the local html version of the LDO, are automatically updated periodically to maintain consistency with the underlying metadata. The index document is kept in a “well known” location, typically at the top level of a collection hierarchy.

This technique addresses a number of related issues. “Easy use” comes from being able to “manually” browse a complete index that is a persistent Web document whose entries contain enough information so that:

- visual browsing is meaningful;
- Web search engines can pickup this document and index the LDO keywords;
- the URL is visible to the human user so that it can be “cut and pasted” from the Web browser interface.

4.2.1 URNs and Persistent Web Server Names

One issue of persistence that is not addressed by the techniques described above is that of naming the Web server where a collection is located (and a server, therefore, that can respond to a query like (7)).

The LDO approach is intended to be consistent with the URN concept ([16]). URNs have two aspects: naming and resolution [22]. Naming must conform to a scheme that ensures uniqueness. Resolution is the process of locating a system (resolver) that can return the location of a server capable of providing (among other things) the URL for the named resource.

For example, the URN for an LDO can include the OID and any of the metadata (e.g. the keywords), plus the LDO-server name. A URN can provide a location independent and globally persistent “name” for the LDO. When the URN is resolved, it can return the current server where the LDO is located. While the URN never changes, what it resolves to will change, when, e.g., it is necessary to update the meta-data, or if the LDO collection home Web server changes.

In the case of LDOs, in order to acquire a resource we need the name of the Web server that can resolve an OID to a resource / object. While in general one may wish to provide a URN for every OID, if LDOs are organized into collections that are constrained to reside on single systems (though there may be multiple instances of these single systems) — in other words, a sub-collection is not split across several Web servers — then assigning URNs for the collection name is sufficient. In this circumstance, given the collection name, resolving the corresponding URN will return the name(s) of servers that can provide the OID-labeled resource. (This ignores the issue discussed in the design document noted below — of moving an LDO from one collection to another. URNs at the collection level also assume that you know the collection that contains the object of interest.)

In principle, several URN mechanisms “exist.” In one approach [4], the Domain Name System can be used to locate servers that can resolve names in a given name space. (So the `collection_name` identifies a collection of LDOs in terms of a `collection_name_namespace`.) Web servers could be established that responded to a query of the form “what are the DNS names of the Web servers that have this collection”.

In a second approach [1] the “Handle” resolver system can directly provide the answer to the query above, as well as supplying other information about the collection, given the collection URN. In this case the URN might look like `urn:hdl:LDO:collection_name` and the Handle System would return a list of Web Servers that could directly satisfy OID-based requests for LDOs in that collection. The uniqueness of collection names comes from the Handle System concept of hierarchical naming authorities.

4.3 Class and Object Structure

More information on the class and object structure, tag-field syntax, etc., may be found at `http://www-itg.lbl.gov/WALDO` .

5 Security

5.1 Security Model

The WALDO security model is intended to provide distributed management of the object use-conditions, as well as distributed access control for the users. (See [10].) WALDO security involves four entities. The Web Certificate Authority agent (e.g., the WebMaster) that manages the server resources and acts as a trusted third-party that provides the security services. The class collection owner who defines the class template and grants read/write permission for objects. The curators who are the people granted write permission and can add, delete, and modify certain class tags and object fields. And users of the information who are typically granted read-only access. Access control methods can be specified on a per-LDO basis, or (the typical case) specified as part of the class template, and inherited by individual LDOs.

The access control methods protect extra-LDO data (i.e., data referred to by links within the LDO) with the same mechanisms as intra-LDO data since access to extra-LDO data is mediated by access to the LDO. Protection of the extra-LDO data from non-LDO access is affected only if that data is behind an access control gateway with the same policies as the LDO access control.

Currently two access control methods are provided: the `.htaccess` mechanism, with or without user authentication, and a public-key infrastructure (PKI) and certificate based access control.

The `.htaccess` access control operates in the usual way, and involves organizing LDOs in Web server directories that have access control lists associated with them.

The PKI access control method uses public-key certificates issued by the collection owner, potentially on a per object basis, to provide general use-conditions. At the moment, this mechanism is primarily used to allow a collection owner (potentially remote from the WALDO Web server) to specify and post (e.g. on a local certificate server) access control lists (ACLs) in the form of signed certificates. These certificates are used as described below to provide strong access control for LDOs. The use-condition certificates that specify access rights will provide a condition on the user identity (e.g., individual name, group or organization membership) and the type of access permitted (e.g., read or write a class/collection, sub-collection, object, or object field).

While complex schemes can be imagined and implemented within this general framework, we have currently defined mechanisms for the following (which is based on our experience in the LBNL Image Library). Objects can inherit their access conditions from the collection class template.

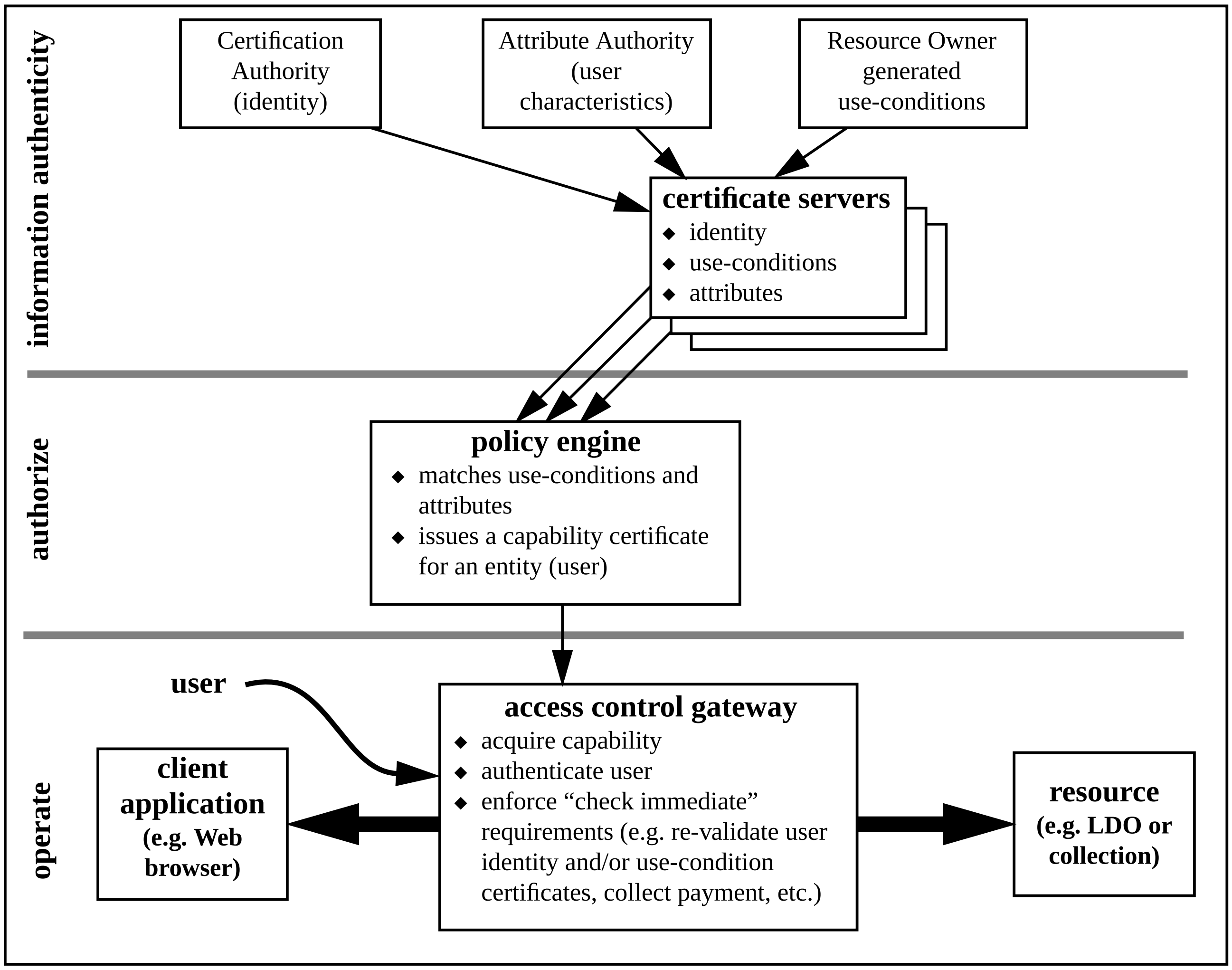

information authenticity
Certification Authority (identity)
Attribute Authority (user characteristics)
Resource Owner generated use-conditions
certificate servers
◆ identity
◆ use-conditions
◆ attributes
authorize
policy engine
◆ matches use-conditions and attributes
◆ issues a capability certificate for an entity (user)
operate
user
access control gateway
◆ acquire capability
◆ authenticate user
◆ enforce "check immediate" requirements (e.g. re-validate user identity and/or use-condition certificates, collect payment, etc.)
client application (e.g. Web browser)
resource (e.g. LDO or collection)

Figure 14. The Use-Condition based Security Architecture.

This allows for ACLs that refer to an entire collection. On a per-object basis, the class access conditions can be overridden by setting `inherit_accessability` to `no_inherit`, and specifying the `Class_owner`, etc., tags, in which case these objects require explicit permission to access. The same holds for individual tag-fields in the object. Access rights are restrictive — if the access specification of a tag is delegated, then specific permission is required for any access to that tag field. This allows restricting access to certain of the LDO components. Again, this may be inherited from the class security prototype, or specified on a per-object basis. Only the collection owner may define the access condition fields, but specific access granting may be delegated. The general approach of an object inheriting accessibility puts an additional burden of correctness on the access control gateway, but is essential for ease of managing large collections.

The relationships of the major elements of the security architecture are illustrated in Figure 14.

Policy-based confidentiality of data is an important aspect of medical, commercial, and some scientific information systems in shared environments. The long term goal of our security architecture work is to provide access control based on owner specified use conditions rather than explicit enumeration of ACLs (which are a special case of use-conditions.) The status of the general approach may be found at `http://www-itg.lbl.gov/security` . This approach is based on the emerging security services for distributed enterprise in both the commercial and scientific communities. (See [10].)

6 Current Experience, Status, and Conclusions

The WALDO large-data-object system has been developed and used at LBNL since early 1995. It currently manages three major collections, each with a different use and curator. The original collection is a scientific image database that supports a small research group. This collection now has over 1400 1–4MByte scanning electron microscope images indexed and described as LDOs. This research group routinely uses WALDO to locate, view, and manage their image collection. They also use WALDO to easily share images with colleagues at other institutions.

The second collection is the cardio-angiography video data repository, and was built in support of a research and development collaboration between Kaiser Permanente Division of Research and Lawrence Berkeley National Laboratory. In this project we have been gathering data on a daily basis since June, 1996. We currently have about 60 patient cardio-angiography studies consisting of about 800 individual video clips stored on the DPSS plus an additional 140 studies that are only on the Mass Storage System. All this data has been automatically captured, sent across a wide-area ATM network, catalogued and entered into WALDO with minimal human intervention. The capture process recovers well from short-term (less than 12 hour) network outages. In this application by virtue of a cache located at the instrument, WALDO's management of on-line data is being evaluated as an alternative to the current hospital use of film. The ability of physicians to view the videos without having to travel to the San Francisco Cardiology facility where the films are stored is a key advantage. However, any network or server failures of even 1 or 2 minutes duration at the time a physician wishes to view a study is a serious inconvenience. Most of the patient-specific associated textual data about the images is kept in restricted-access hospital data bases, and images are referenced by having URLs pointing to a WALDO server.

The third collection is a on-line digital photo archive. We currently have about 700 images on-line selected from the hundreds of thousands of photos in Berkeley Lab's archive. These on-line photos are used to support a variety of Web publications (e.g., [8]) and for general browsing or research by people within and outside of the Lab.

The concept of real-time collection of data from on-line instrument systems, followed by automatic cataloguing that results in an elementary, but useful object store, is quite powerful. It allows immediate meaningful access to large classes of data that can undergo refinement over time as the knowledge about the data increases. A high-speed, network-based distributed cache plays a central role in our approach, providing storage for buffering incoming data, intermediate processing, and as an application cache. An experimental security architecture provides data-owner defined, per-object access control, and is intended to explore the issues and capabilities for on-line and widely available, but strongly protected, sensitive information.

7 Acknowledgments

The overall approach based on wide-area network distributed high-speed caches has evolved over several years with funding from the Mathematical, Information, and Computational Sciences Division of the Office of Energy Research, U. S. Department of Energy, and DARPA Information Technology Office (the DPSS), and from the Lawrence Berkeley National Laboratory all of which are gratefully acknowledged.

In the Kaiser CalREN project, E-J. Pol of Philips Research, Palo Alto, CA, designed and implemented the workstation interface for the cardio-angiography system. Dr. J. Terdiman, of the Kaiser Permanente Division of Research provided the leadership for the health care aspects of the project. Dr. Bob Lundstrum, Kaiser San Francisco Hospital, Cardiac Catheterization Laboratory, provided guidance and information specific to the health care application. Pacific Bell, through its CalREN program, provided funding for originally connecting the LBNL, Kaiser, and Philips sites to the SF Bay Area ATM network. Sun Microsystems assisted in various ways, including loaning equipment to several Kaiser sites.

Mary Thompson directs the WALDO project, Brian Tierney directs the DPSS project, and Case Larsen is the implementer of the underlying security architecture. The LungLab collection is a collaboration with Jacob Bastacky, Lung Microscopy Group, Molecular and Nuclear Medicine Department, Life Sciences Division of Lawrence Berkeley National Laboratory

Finally, one of us (Johnston) would like to acknowledge Ace Allen, M.D., of the KU Medical Center in Kansas City for enthusiastic early support of the idea of networked medical information systems; Ms. Sue Kwentus, RN, and the MCC HOST/OSL group, for focusing some of the early ideas; and Sprint generally, and Lt. Col. John Strand, U.S. Army (ret.), now Director of Technology Planning and Integration at Sprint, in particular, for support of the MAGIC backbone network (without which this work probably would not have been done) (See [14]).

8 References

[1] W. Arms, D. Ely. "The Handle System", An Internet Engineering Task Force, Uniform Resource Identifiers working group draft available at `http://www.ietf.cnri.reston.va.us/ids.by.wg/uri.html`

[2] T. Bray, R. V. Guha, "An MCF Tutorial." `http://www.w3.org/TR/NOTE-MCF-XML/MCF-tutorial.html`

[3] The Distributed-Parallel Storage System (DPSS) Home Page (`http://www-itg.lbl.gov/DPSS`)

[4] R. Daniel, M. Mealling. "Resolution of Uniform Resource Identifiers using the Domain Name System", An Internet Engineering Task Force, Uniform Resource Names working group draft available at `http://www.ietf.cnri.reston.va.us/ids.by.wg/urn.html`

[5] Fuller, B., I. Richer "The MAGIC Project: From Vision to Reality," IEEE Network, May, 1996, Vol. 10, no. 3.

[6] Manber, U. and S. Wu, "GLIMPSE: A Tool to Search Through Entire File Systems". University of Arizona Computer Science Technical Report TR 93-34. Available at `http://glimpse.cs.arizona.edu`.

[7] Greiman, W., W. E. Johnston, C. McParland, D. Olson, B. Tierney, C. Tull, "High-Speed Distributed Data Handling for HENP". International Conference on Computing in High Energy Physics, Berlin, Germany, April, 1997. Also available at `http://www-itg.lbl.gov/STAR` .

[8] J. L. Heilbron, Robert W. Seidel, Bruce R. Wheaton, "Lawrence and His Laboratory: A Historian's View of the Lawrence Years." `http://www.lbl.gov/Science-Articles/Research-Review/Magazine/1981/index.html`

[9] See `http://www-itg.lbl.gov/ImgLib/ImgLib_intro.html`

[10] Johnston, W. and C. Larsen, "Security Architectures for Large-Scale Remote Collaboratory Environments: A Use-Condition Centered Approach to Authenticated Global Capabilities" (draft at `http://www-itg.lbl.gov/security/publications.html`)

[11] W. Johnston, and D. Agarwal, "The Virtual Laboratory: Using Networks to Enable Widely Distributed Collaboratory Science" A NSF Workshop Virtual Laboratory whitepaper. (See `http://www-itg.lbl.gov/~johnston/Virtual.Labs.html`)

[12] Kaiser - LBNL - Philips CalREN project. See `http://www-itg.lbl.gov/Kaiser/LKP`.

[13] "TerraVision: A Terrain Visualization System". Y. Leclerc and S. Lau, Jr. SRI International, Technical Note #540, Menlo Park, CA, 1994. Also see: `http://www.ai.sri.com/~magic/terravision.html`

[14] MAGIC (Multidimensional Applications and Gigabit Internetwork Consortium) is a gigabit network testbed that was established in June 1992 by the U. S. Government's Advanced Research Projects Agency (ARPA). The testbed is a collaboration between LBNL, Minnesota Supercomputer Center, SRI, Univ. of Kansas, Lawrence, KS, USGS - EROS Data Center, CNRI, Sprint, U. S. West, Southwest Bell, and Splitrock Telecom. More information about MAGIC may be found on the WWW home page at: `http://www.magic.net`

[15] "National Transparent Optical Network Consortium". See `http://www.ntonc.org` . (NTONC is a program of collaborative research, deployment and demonstration of an all-optical open testbed communications network.)

[16] "Functional Requirements for Uniform Resource Names". K. Sollins, L. Masinter. Internet Engineering Task Force, Request for Comment no. 1737. December, 1994. Available at `http://ds.internic.net/rfc/rfc1737.txt` .

[17] "Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types". N. Freed and N. Borenstein.

Internet Engineering Task Force, Request for Comment no. 2046. November, 1996. http://info.internet.isi.edu:80/in-notes/rfc/files/rfc2046.txt

[18] J. Palme and A. Hopmann, “MIME E-mail Encapsulation of Aggregate Documents, such as HTML (MHTML).” ftp://ds.internic.net/rfc/rfc2110.txt

[19] E. Levinson, “Content-ID and Message ID Uniform Resource Locators.” ftp://ds.internic.net/rfc/rfc2111.txt

[20] E. Levinson, “The MIME Multipart / Related Content-type.” ftp://ds.internic.net/rfc/rfc2112.txt

[21] Tierney, B., W. Johnston, G. Hoo, J. Lee, “Performance Analysis in High-Speed Wide-Area ATM Networks: Top-to-Bottom End-to-End Monitoring”, IEEE Network, May, 1996, Vol. 10, no. 3. LBL Report 38246, 1996. (Also see http://www-itg.lbl.gov/DPSS/papers.html .)

[22] “Uniform Resource Names: A Progress Report”, The URN Implementer Group. D-Lib Magazine, February 1996 (http://www.dlib.org/dlib/february96/02arms.html)

LBNL-39613
UC-405

Real-Time Digital Libraries Based on Widely Distributed, High Performance Management of Large-Data-Objects

William Johnston, Jin Guojun, Case Larsen,
Jason Lee, Gary Hoo, Mary Thompson,
Brian Tierney, and Joseph Terdiman, M.D.

August, 1997